# UNITED STATES PATENT AND TRADEMARK OFFICE

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/583,747 | 05/31/2000 | Harlan Sexton | 50277-450 | 4124 |

7590          05/05/2004

DITTHAVONG & CARLSON, P. C.
10507 BRADDOCK ROAD
SUITE A
fairfax, VA  22032

| EXAMINER |
|---|
| NAHAR, QAMRUN |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2124 | |

DATE MAILED: 05/05/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

| Office Action Summary | Application No. 09/583,747 | Applicant(s) SEXTON, HARLAN | |
|---|---|---|---|
| | Examiner Qamrun Nahar | Art Unit 2124 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

## Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE **3** MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

## Status

1)☒ Responsive to communication(s) filed on *17 March 2004*.

2a)☒ This action is **FINAL**.     2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

## Disposition of Claims

4)☒ Claim(s) *1-22* is/are pending in the application.

   4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-22* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

## Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

   Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

   Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

## Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   a)☐ All   b)☐ Some * c)☐ None of:

   1.☐ Certified copies of the priority documents have been received.

   2.☐ Certified copies of the priority documents have been received in Application No. _____.

   3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

   * See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1)☐ Notice of References Cited (PTO-892)
2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3)☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5)☐ Notice of Informal Patent Application (PTO-152)
6)☐ Other: _____.

## DETAILED ACTION

1.    This action is in response to the amendment filed on 3/17/04.

2.    Claims 1-22 are pending.

3.    Claims 1-18 stand finally rejected under 35 U.S.C. 102 (b) as being anticipated by

Arsenault (U.S 5,408,650).

4.    Claims 19-22 stand finally rejected under 35 U.S.C. 103(a) as being unpatentable over

Arsenault (U.S 5,408,650) in view of Elliott (U.S. 4,945,474).

### *Claim Rejections - 35 USC § 102*

5.    The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the

basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6.    Claims 1-18 are rejected under 35 U.S.C. 102 (b) as being anticipated by Arsenault (U.S

5,408,650).

### Per Claim 1:

Arsenault discloses a method of analyzing a program (See col.2, li.61-66) comprising the

steps of logging a plurality of stack traces and respective tags in a log file at respective points

during execution of the program (column 6, lines 2-10 and lines 50-53; and see Fig. 2, item 26;

There are more than one stack traces, only one is displayed at a time. The log file is displayed on

the screen. Arsenault inherently teaches a log file because there is a debugger, which stores the

information it collects in a log file (column 3, lines 37-45). Debuggers write stack traces and

other debugging output to a *log file*. That is, debuggers need to store the information it collects

in a *log file* for later processing. Arsenault teaches that "the memory analysis system

communicates with a debugger. The debugger analyzes the execution of the program in a

conventional manner and relates 'symbols' and locations of the compiled execution version of

the program, that is, the image file, to symbols and locations in the source code version of the

program. The memory analysis system then uses this *information* to generate the call-stacks and

make available the related lines of source code."; see column 3, lines 37-45 (emphasis added).

Therefore, the debugger stores this information in a *log file* for later processing); and recording

within the log file one or more of the tags as one or more marked tags ("a creation count '215'

indicates that this is the 215$^{th}$ block allocated to the program; the **address** '00126398' of the

location at the beginning of the segment; the number of locations '00001410' in the segment; the

**address** '001277A8' of the location at the end of the segment; and the name of the segment type

'EF_BLOCK_MANAGER,' which is the name of an associated program sub-routine" in column

6, lines 41-49 (emphasis added) and see Figure 2, item 27; "creation count" is interpreted as

"marked tags".).


**Per Claim 2:**

Arsenault further discloses producing a report based on log file (col.2, li.64-66; col.5,

li.67 to col.6, li.16; fig.2, ref. 28).


**Per Claim 3:**

Arsenault further discloses identifying one or more of the stack traces associated with any of the one or more marked tags; and producing said report based on the identified stack traces (col.3, li.11-22).

## Per Claim 4:

Arsenault further discloses identifying the last stack trace associated with any of the one or more marked tags; and producing said report based on the identified stack traces (col.3, li.11-22; col.11, li.8-12). For a finite quantity of identified stack traces, the feature is inherent in Arsenault's system to enable specific reports to be generated for any of the identified stack traces (col.12, li.6-8).

## Per Claim 5:

Arsenault further discloses a method wherein the tags indicate respective addresses of allocated objects (col.6, li.41-9); and the one or more marked tags indicate one or more respective addresses of migrated objects (col.6, li.50-67 to col.7, li.1-4; deallocation routine indicates migrated objects; that is, objects that have been deallocated are migrated objects. Allocated objects that have been deallocated are objects that have migrated from their current memory locations. Therefore, the one or more marked tags indicate one or more respective addresses of migrated objects, that is, addresses of allocated objects that have been migrated.).

## Per Claim 6:

Arsenault discloses a method for producing a diagnostic report for a program comprising

accessing a log file comprising a list of stack traces and respective tags at associated points

during execution of the program and comprising one or more marked tags ("a creation count

'215' indicates that this is the 215th block allocated to the program; the **address** '00126398' of

the location at the beginning of the segment; the number of locations '00001410' in the segment;

the **address** '001277A8' of the location at the end of the segment; and the name of the segment

type 'EF_BLOCK_MANAGER,' which is the name of an associated program sub-routine" in

column 6, lines 41-49 (emphasis added); column 6, lines 2-5 and lines 30-40; see Figure 2, items

26 and 27; "creation count" is interpreted as "marked tags". Furthermore, Arsenault inherently

teaches a log file because there is a debugger, which stores the information it collects in a log file

(column 3, lines 37-45). Debuggers write stack traces and other debugging output to a *log file*.

That is, debuggers need to store the information it collects in a *log file* for later processing.

Arsenault teaches that "the memory analysis system communicates with a debugger. The

debugger analyzes the execution of the program in a conventional manner and relates 'symbols'

and locations of the compiled execution version of the program, that is, the image file, to

symbols and locations in the source code version of the program. The memory analysis system

then uses this *information* to generate the call-stacks and make available the related lines of

source code."; see column 3, lines 37-45 (emphasis added). Therefore, the debugger stores this

information in a *log file* for later processing); and producing the diagnostic report based on the

log file (column 7, lines 61-67 to column 8, lines 1-39; Figure 2, items 28 and 36).


**Per Claim 7:**

Arsenault further discloses identifying one or more of the stack traces associated with any

of the one or more marked tags; and producing said report based on the identified stack traces

(col.3, li.11-22).

## Per Claim 8:

Arsenault further discloses identifying the last stack trace associated with any of the one

or more marked tags; and producing said report based on the identified stack traces (col.3,

li.11-22; col.11, li.8-12). For a finite quantity of identified stack traces, the feature is inherent in

Arsenault's system to enable specific reports to be generated for any of the identified stack traces

(col.12, li.6-8).

## Per Claim 9:

Arsenault further discloses a method wherein the tags indicate respective addresses of

allocated objects (col.6, li.41-9); and the one or more marked tags indicate one or more

respective addresses of migrated objects (col.6, li.50-67 to col.7, li.1-4; deallocation routine

indicates migrated objects; that is, objects that have been deallocated are migrated objects.

Allocated objects that have been deallocated are objects that have migrated from their current

memory locations. Therefore, the one or more marked tags indicate one or more respective

addresses of migrated objects, that is, addresses of allocated objects that have been migrated.).

## Per Claims 10-14:

These claims represent an apparatus performing a method corresponding to the methods

of claims 1-5, respectively. The claims are rejected under the same arguments as cited above,

with Column 3, Lines 23 to 36 referencing the apparatus (computer-readable medium bearing

instructions for analyzing a program).

**Per Claims 15-18:**

These claims represent an apparatus performing a method corresponding to the methods

of claims 6-9, respectively. The claims are rejected under the same arguments as cited above,

with Column 3, Lines 23 to 36 referencing the apparatus (computer-readable medium bearing

instructions for analyzing a program).

*Claim Rejections - 35 USC § 103*

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

8. Claims 19-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Arsenault

(U.S 5,408,650) in view of Elliott (U.S. 4,945,474).

**Per Claim 19:**

The rejection of claim 4 is incorporated, and Arsenault further teaches producing the

report (fig.2, ref.26; col.6, li.30-33). Arsenault does not explicitly teach processing the log file

from the end backward until the beginning. Elliott teaches processing the log file from the end

backward until the beginning (fig.5, ref.276; col.6, li.64-68 to col.7, li.1-5).

It would have been obvious to one having ordinary skill in the computer art at the time of

the invention was made to modify the method disclosed by Arsenault to include processing the

log file from the end backward until the beginning using the teaching of Elliott. The

modification would be obvious because one of ordinary skill in the art would be motivated to

process the newest entry to a sequentially-generated log file first, as taught by Elliott (col. 6,

li.64-68).

## Per Claim 20:

This is another version of the claimed method discussed above, claim 19, wherein all

claim limitations also have been addressed and/or covered in cited areas as set forth above.

Thus, accordingly, this claim is also obvious.

## Per Claims 21 and 22:

These are computer-readable medium versions of the claimed method discussed above,

claim 19, wherein all claim limitations also have been addressed and/or covered in cited areas as

set forth above. Thus, accordingly, these claims are also obvious.

### *Response to Arguments*

9.     Applicant's arguments filed on 3/17/04 with respect to claims 1-22 have been fully

considered but they are not persuasive.

*In the remarks, the applicant argues that:*

a)       A. CLAIMS 5, 9, 14, AND 18 ARE NOT ANTICIPATED BECAUSE *ARSENAULT*

FAILS TO DISCLOSE "MIGRATED OBJECTS."

To anticipate a patent claim, every element and limitation of the claimed invention must

be found in a single prior art reference, arranged as in the claim. Karsten Mfg. Corp. v.

Cleveland Golf Co., 242 F.3d 1376, 1383, 58 USPQ2d 1286, 1291 (Fed. Cir. 2001); Scripps

Clinic & Research Foundation v. Genentech, Inc., 927 F.2d 1565, 1576, 18 USPQ2d 1001, 1010

(Fed. Cir. 1991).

The rejection of claims 5, 9, 14, and 18 over Arsenault is improper because the applied

reference does not disclose the limitations of the claims. For example, claims 5, 9, 14, and 18

recite:

> 5. (Previously Presented) The method according to claim 1, wherein: the tags indicate
> respective addresses of allocated objects; and the one or more marked tags indicate one or
> more respective addresses of migrated objects.
>
> 9. (Previously Presented) The method according to claim 6, wherein: the tags indicate
> respective addresses of allocated objects; and the one or more marked tags indicate one or
> more respective addresses of migrated objects.

Claims 14 and 18 are computer-readable medium claims corresponding to claims 5 and 9,

respectively. These elements are not disclosed in Arsenault.

Arsenault does not have any disclosure of "migrated objects" as recited in claims 5, 9, 14,

and 18. The Examiner's response on page 10 of the Office Action states that "deallocation

routine indicates migrated objects; that is, objects that have been deallocated are migrated

objects." However, "migrated" and "deallocated" are clearly different: "migrated" means

"moved," but "deallocated" means "stopped existing."

Moreover, claims 5, 9, 14, and 18 recite that the "the one or more **marked tags indicate** one or more respective **addresses** of migrated objects" (emphasis added). Unless the patent otherwise provides, a claim term cannot be given a different meaning in the various claims of the same patent. In the rejection of parent claim 1, the "marked tag" is read on a creation count, an ordinal number (col. 6:42), which does not indicate an address at all, let alone an address of a migrated object. The newly cited cols. 6:50-7:4 do not support the rejection, however, because it fails to discuss creation counts at all. Thus, there is no teaching of marked tags indicating "one or more respective **addresses** of migrated objects," as recited in claims 5, 9, 14, and 18.

*Examiner's response:*

a)     Examiner strongly disagrees with applicant's assertion that Arsenault fails to disclose the claimed limitations recited in claims 5, 9, 14, and 18. Arsenault clearly shows each and every limitation in claims 5, 9, 14, and 18. Arsenault discloses that "the tags indicate respective addresses of allocated objects" ("a creation count '215' indicates that this is the 215th block allocated to the program; the **address** '00126398' of the location at the beginning of the segment; the number of locations '00001410' in the segment; the **address** '001277A8' of the location at the end of the segment; and the name of the segment type 'EF_BLOCK_MANAGER,' which is the name of an associated program sub-routine" in column 6, lines 41-49 (emphasis added) and see Figure 2, item 27; "creation count" is interpreted as "marked tags".); and "the one or more marked tags indicate one or more respective addresses of migrated objects" (column 6, lines 50-67 to column 7, lines 1-4; deallocation routine indicates migrated objects; that is, objects that have been deallocated are migrated objects. Allocated

objects that have been deallocated are objects that have migrated from their current memory

locations. Therefore, the one or more marked tags indicate one or more respective addresses of

migrated objects, that is, addresses of allocated objects that have been migrated.). In addition,

see the rejection above in paragraph 6 for rejection to claims 5, 9, 14, and 18.

*In the remarks, the applicant argues that:*

b)      B. ARSENAULT FAILS TO ANTICIPATE CLAIMS 1-18 BECAUSE ARSENAULT

DISCLOSES NEITHER "LOGGING A PLURALITY OF STACK TRACES ... IN A LOG

FILE" NOR "ACCESSING A LOG FILE COMPRISING A LIST OF STACK TRACES."

Turning now to the rejection of all claims 1-18, this rejection is respectfully traversed

because Arsenault does not disclose the limitations recited in independent claims 1, 6, 10, and

15. For example, independent claims 1 and 10 recite "logging a plurality of stack traces and

respective tags in a log file," and independent claims 6 and 15 recite "accessing a log file

comprising a list of stack traces and respective tags."

Arsenault contains no explicit disclosure of a "log file" and at best discloses a

representation displayed to the user on the screen of a display device that includes "a listing 26 of

the call-stack associated with a selected memory segment" (col. 6:2-4, note singular

"call-stack"). Specifically, Arsenault discloses a graphic representation of a map of allocated

memory segments depicted by segment type and various listings shown on a display device to a

user (cols. 5:65-6:4), but not the "recording within the log file one or more of the tags as one or

more marked tags" as presently recited in independent claims 1 and 10 and "accessing a log file

comprising a list of stack traces and respective tags."

In response to Applicant's arguments in the Appeal Brief, the Office Action, p. 13,

resorted to inherency to fill out the disclosure missing in Arsenault: "Arsenault inherently

teaches a log file because there is not a debugger without a log file, see column 3, lines 37-45."

However, col. 3:37-45 has nothing to do with log file, but with accessing the " 'symbols' and

locations of the **compiled executable version** of the program" (col. 3:40-41, emphasis added). A

compiled executable version of a program is not a program, and, even if it were, debuggers do

not write stack traces and other debugging output to compiled executable versions of programs.

Furthermore, the rejection does meet the standard imposed by the Federal Circuit in using

inherency. Although "inherency places subject matter in the public domain as well as express

disclosure," Schering Corp. v. Geneva Pharms., Inc., No. 02-1540 (Fed. Cir., August 1, 2003),

slip op. at 9, it must be clear that the missing descriptive matter is necessarily present in the

reference to establish inherency. In re Roberston, 49 USPQ2d 1949, 1951 (Fed. Cir., 1999).

Under the principles of inherency, the prior art must necessarily function in accordance with, or

include, the claim limitations. MEHL/Biophile Int'l., 52 USPQ2d 1303 (Fed. Cir. 1999); see also

Schering Corp., id. at 8 ("necessarily and inevitably"). As explained in MEHL/Biophile Int'l:

> Inherency, however, may not be established by probabilities or possibilities. The mere
> fact that a certain thing may result from a given set of circumstances is not sufficient. If,
> however, the disclosure is sufficient to show that the natural result flowing from the
> operation as taught would result in the performance of the questioned function, it seems
> to be well settled that the disclosure should be regarded as sufficient.

The attempt to base the rejection on something other than what Arsenault teaches follows

far short of these requirements for inherency. For one, there is no evidence on the record that

"there is not a debugger without a log file." However, it is not a necessary requirement for

debuggers to have log files, especially on microprocessor systems. If this statement is "well-

known," then it is traversed and the Patent Office is requested to articulate and place on the

record the "common knowledge" used to negate patentability. In re Zurko, 258 F.3d 1379, 1386,

59 USPQ2d 1693, 1697 (Fed. Cir. 2001);, In re Swig Su Lee, No. 00-1158 (Fed. Cir., Jan. 18,

2002). As another example, even if some evidence were to be shown of a debugger using a log,

one must still show a debugger that logs stack traces, because debuggers do not necessarily log

everything.

*Examiner's response:*

b)      Examiner strongly disagrees with applicant's assertion that Arsenault fails to disclose the

claimed limitations recited in independent claims 1, 6, 10, and 15. Arsenault clearly shows each

and every limitation in independent claims 1, 6, 10, and 15.

Arsenault discloses "logging a plurality of stack traces and respective tags in a log file at

respective points during execution of the program" (column 6, lines 2-10 and lines 50-53; and

see Fig. 2, item 26; There are more than one stack traces, only one is displayed at a time. The log

file is displayed on the screen. The Examiner would like to point out that the log file is not

interpreted as a "screen". Arsenault inherently teaches a log file because there is a debugger,

which stores the information it collects in a log file, see column 3, lines 37-45); and recording

within the log file one or more of the tags as one or more marked tags ("a creation count '215'

indicates that this is the 215th block allocated to the program; the **address** '00126398' of the

location at the beginning of the segment; the number of locations '00001410' in the segment; the

**address** '001277A8' of the location at the end of the segment; and the name of the segment type

'EF_BLOCK_MANAGER,' which is the name of an associated program sub-routine" in column

6, lines 41-49 (emphasis added) and see Figure 2, item 27; "creation count" is interpreted as

"marked tags".) as recited in claims 1 and 10. In addition, Arsenault discloses "accessing a log

file comprising a list of stack traces and respective tags at associated points during execution of

the program and comprising one or more marked tags" (column 6, lines 2-5 and lines 30-45;

Figure 2, item 26; "creation count" is interpreted as "marked tags"); and "producing the

diagnostic report based on the log file" (column 7, lines 61-67 to column 8, lines 1-39; Figure 2,

items 28 and 36) as recited in claims 6 and 15.

The applicant argues that "**A compiled executable version of a program is not a**

**program**, and, even if it were, debuggers do not write stack traces and other debugging output to

compiled executable versions of programs." By definition, a program is a sequence of

instructions that can be executed by a computer. The term can refer to the original source code

or to the executable (machine language) version. The Examiner is unclear as to what applicant is

trying to point out with this contradictory statement. Furthermore, debuggers do not write stack

traces and other debugging output to compiled executable versions of programs. Debuggers

write stack traces and other debugging output to a *log file*. That is, debuggers need to store the

information it collects in a *log file* for later processing. Arsenault teaches that "the memory

analysis system communicates with a debugger. The debugger analyzes the execution of the

program in a conventional manner and relates 'symbols' and locations of the compiled execution

version of the program, that is, the image file, to symbols and locations in the source code

version of the program. The memory analysis system then uses this *information* to generate the

call-stacks and make available the related lines of source code."; see column 3, lines 37-45

(emphasis added). Therefore, the debugger stores this information in a *log file* for later

processing.

Furthermore, see the rejection above in paragraph 6 for rejection to independent claims 1,

6, 10, and 15.

*In the remarks, the applicant argues that:*

c)      C. THERE IS NO MOTIVATION TO COMBINE ARSENAULT AND ELLIOT ET AL.

FOR CLAIMS 19-22.

The obviousness rejection of claims 19-22 is respectfully traversed because there is no

motivation to combine Arsenault and Elliott et al. As best understood, the Office Action attempts

to read the recited "log file" on an inherent log file in debuggers. However, as explained above,

this use of inherency is insufficient as a matter of law, because not all debuggers use log files and

there is no reference on the record that even states that all debuggers use log files. Even if the

Examiner were to find a reference explicitly disclosing this, there is still no motivation for one of

ordinary skill in the art to modify debugger log files that log output with the non-analogous

Elliott et al. because Elliott et al. is not directed at all to debugger log files but to processing

relational database recovery logs. The formats are different, and all the special processing in

Elliott et al. to handle 1/O errors and system crashes when logging database transactions is

irrelevant to Arsenault, which does not use database transactions.

*Examiner's response:*

c)     The Examiner has already addressed the applicant's arguments regarding "log file" in the

Examiner's Response (b) above.

In response to applicant's argument that Elliott is nonanalogous art, it has been held that a

prior art reference must either be in the field of applicant's endeavor or, if not, then be

reasonably pertinent to the particular problem with which the applicant was concerned, in order

to be relied upon as a basis for rejection of the claimed invention.  See *In re Oetiker*, 977

F.2d 1443, 24 USPQ2d 1443 (Fed. Cir. 1992).  In this case, Elliott teaches processing log files

from the end backward until the beginning (fig.5, ref.276; col.6, li.64-68 to col.7, li.1-5).

Therefore, Elliot is analogous art.

The Examiner pointed out in the previous Office Action, Paper No. 17, why it would

have been obvious to combine Arsenault and Elliott, that is, the modification would be obvious

because one of ordinary skill in the art would be motivated to process the newest entry to a

sequentially-generated log file first, as taught by Elliott (col. 6, li.64-68).  The applicant has

failed to point out any error in the motivation in the rejection of claims 19-22.  Therefore, the

rejection is maintained.

*Conclusion*

10.    **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time

policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE

MONTHS from the mailing date of this action.  In the event a first reply is filed within TWO

MONTHS of the mailing date of this final action and the advisory action is not mailed until after

the end of the THREE-MONTH shortened statutory period, then the shortened statutory period

will expire on the date the advisory action is mailed, and any extension fee pursuant to 37

CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

however, will the statutory period for reply expire later than SIX MONTHS from the mailing

date of this final action.

11.     Any inquiry concerning this communication from the examiner should be directed to

Qamrun Nahar whose telephone number is (703) 305-7699. The examiner can normally be

reached on Mondays through Thursdays from 9:00 AM to 6:30 PM. The examiner can also be

reached on alternate Fridays.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Kakali Chaki, can be reached on (703) 305-9662. The fax phone number for the

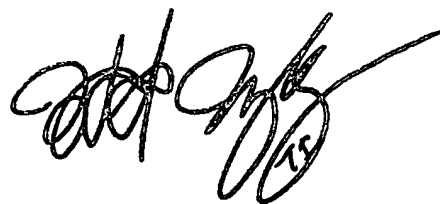organization where this application or processing is assigned is (703) 872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding

should be directed to the receptionist whose telephone number is (703) 305-3900.

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

QN

April 19, 2004

TODD INGBERG
PRIMARY EXAMINER